

ATARI
COMPUTER
ENTHUSIASTS

3662 Vine Maple Dr. Eugene OR 97405

JUNE 1982

Mike Dunn & Jim Bumpas, Editors



.....OOPS!!

There are a few errors I found in the schematic for the cassette interface. The unmarked resistor between the two 820 pF caps and op amp pin 2 is 240K. In the same position off pin 6 is an 330k resistor. Following the diode connected to pin 1 and pin 7 is a 15k resistor. Anyone have one working yet?

In the RPM article in FORTH, there are two errors in the definition of the FORTH words; one definition has an extra ";" and one has it left out. See if you can find it!

FLYING HIGH WITH ATARI PILOT

By Ruth Ellsworth

Programming and education; kids and computers go together, but until Atari PILOT with "Turtle" Graphics arrived at our home it was not always a successful combination.

We purchased our computer with the idea that we would use it as an educational enrichment tool, and as a means to introduce our children to computer language. We soon found that BASIC was a little harder than we had thought it would be, and was not suitable for our younger children. We also discovered that the educational software we needed and wanted was either unavailable or available only at prices that blew both our minds and our budget.

Having a bad case of "little red henitis," I began the task of trying to put the curriculum we desired into BASIC. It turned out to be a time consuming job for one whose interest in math never extended beyond the boy across the row in algebra class some years ago, and who is frequently interrupted by children of all sizes. I must admit that I was making slow progress until PILOT came along.

The documentation that comes with the Atari PILOT package is excellent and it soon became apparent that this language had been written by and for educators. Atari PILOT makes it possible for even a beginning programmer like myself to easily "translate" any standard teacher's edition of instructional material into an individualized learning package.

I find the ability to write my own educational programs preferable to purchasing those made elsewhere unless I can alter them to suit our needs. The major advantage to being able to create our own educational software is that I can tailor the curriculum to meet the needs of each particular child. I have the option of taking the information from numerous sources and presenting it at the level and speed which fits each one, and the children really respond to the personalized encouragement by name and interest written into a program that is obviously specifically theirs.

The commands used in PILOT are easy to use and remember. I found the Match, Match Jump, Use If Yes, and Use If No particularly useful in programming curriculum. I had never stopped to realize before I began programming in PILOT that most educational curriculum from math time tests to the SAT use the simple approach of presenting information and then requiring students to match the desired answer. Once I came to that obvious conclusion the way the PILOT adapts to curriculum became apparent.

The TRACE:ON and TRACE:OFF options which allow one to follow the program during operation in order to locate any bugs are really nice and greatly add to the ease in which programming can be accomplished with PILOT. I also found the the AUTO numbering, and the REN (renumbering) options very useful and time saving.

"Turtle" Graphics are not only fun, but very fast. The oversized text modes used in the "slide show" on the demo tapes seem to be connected to the graphics. They not only add a colorful and easily read touch to programming, but are especially adaptable to such educational uses as spelling, and math speed drills. The sample program which I am including at the end of this review demonstrates the two modes available and the color options. C:@B1373=16 gives a split screen allowing the use of the usual commands in PILOT, C:@B1374=2 sets the oversize mode (either 1 or 2). Read and Write commands must be used to place oversized text on the screen. I have found it easiest to think of the screen in the oversize text modes as a slide, it does not scroll, and therefore, the Esc. Clear command must be used frequently to avoid error messages. The simple program I am using to demonstrate can be made to run very fast, in fact the PA commands are used specifically to slow it down. In a full screen oversize text mode this little math program will run as fast as you can input the answers without the PA commands and *YES module.

Atari PILOT is not only perfectly suited to educational programming, but it is an excellent introduction to programming for young children. The student manual is extensively illustrated by clever cartoon like pictures, and contains a list and short description of all the commands in the Table of Contents. Our middle boys were delighted with it. By the end of our first session they were writing clever little programs using the Type, Accept, and Match commands. They found "Turtle" Graphics easy enough to understand and were soon able to make simple pictures to add to their programs.

Programming in PILOT has been a pleasant experience for our whole family. It has encouraged our children to experiment with programming and to use our computer as more than a game

machine. I am especially delighted with PILOT as a first computer language because it encourages modular programming which I believe to be good technique and which the younger boys did not seem to grasp as they struggled with BASIC.

In short, I found Atari PILOT to be an adaptable, easily learned language including powerful options, such as the ability to accept machine language programs, which I, as a beginning programmer, do not understand. I believe that PILOT opens the door to the use of the computer as an educational tool not only by educators, but by parents; and I believe that there are many families looking for this type of program and educational enrichment for their children. I would encourage anyone to take off with Atari PILOT, it really gets programming off the ground!

PILOT DEMO PROGRAM

```

10 R:SIMPLE MULTIPLICATION PROGRAM
20 R:By Ruth Ellsworth
30 R:ACE NEWSLETTER
40 R:3662 VINE MAPLE DR.
50 R:EUGENE, OREGON 97405
60 R:REMOVE *YES AND PAUSES TO INCREASE
70 R:SPEED OF PROGRAM
80 C:@N=0 [initialize # of problems to 0
90 C:@R=0 [initialize # right to 0
100 *MULTIPLY [label for main module
110 C:@N=#N+1 [compute total # of problems
120 C:@I=?\13 [generate random # I
130 C:@J=?\13 [generate random # J
140 C:@B1373=16 [split screen with text window
150 C:@B1374=2 [oversize text mode 2
160 WRITE:S,I*#J= [write problem in oversize text on
screen-color yellow
170 C:@X=#I*#J [compute I*#J
180 READ:K,$Y [read variable $Y from keyboard to buffer (it
can be a number)
190 M:@X [test match $Y to #X
200 JY:*YES [jump to module *YES line 280 if $Y matches #X
210 MS:S [test match $Y to S to stop program
220 JY:*COUNTER [jump to module *COUNTER line 380 if $Y
matches S
230 WRITE:S, ) [clear screen
240 WRITE:S,I*#J=#X [writes correct answer on the screen
in oversize text-color yellow-if modules *YES and *COUNTER
were not used
250 PA:90 [pause so that correct answer can be studied
260 WRITE:S, ) [clear screen
270 J:*MULTIPLY [return to main module line 100 for next
problem
280 *YES [module for correct answer (see line 190)
290 WRITE:S, ) [clear screen so problem does not appear twice
300 WRITE:S,I*#J=#X [reinforce correct answer given,
oversize text-yellow
310 WRITE:S,horray for david [inverse small letters-writes
oversize text-color red,personalized encouragement
320 WRITE:S,you did it! [inverse small letters writes
oversize text-color red, more encouragement
330 PA:30 [pause so reinforcement and encouragement may be
read
340 WRITE:S, ) [clear screen
350 C:@R=#R+1 [compute total correct answers
360 J:*MULTIPLY [return to main module line 100 for next
problem
370 E: [end *YES module
380 *COUNTER [module to output progress (see line 220)
390 WRITE:S, ) [clear screen
400 WRITE:S,#N [writes #problems given to screen in
oversize text-color yellow
410 WRITE:S,PROBLEMS [writes the word problems in
oversize text-color yellow
420 WRITE:S,#R [writes the #correct to screen in oversize
text-color yellow
430 WRITE:S,RIGHT [write the word right in oversize
text-color yellow
440 PA:120 [pause so progress can be read
450 WRITE:S, ) [clear screen

```



```

460 WRITE:S,WOULD YOU LIKE [writes inverse capital
letters oversize text-color blue
470 WRITE:S,TO CONTINUE? [writes inverse capital
letters-color blue
480 READ:K,$C [reads $C from keyboard to buffer-the
continue option
490 MS:Y,OK, O.K., SURE, ALRIGHT [match string $C to
continue count
500 WRITE:S, [clear screen
510 JY:*MULTIPLY [jump to main module line 100 if $C
matches
520 WRITE:S,would you like [writes oversize text-color green
530 WRITE:S,to start over? [writes oversize text-color
green, new count option
540 READ:K,$O Ereads $O from keyboard to buffer
550 MS:Y,OK,O.K.,SURE,ALRIGHT [match $O for instructions
to start over
560 JY:*OVER [jump to module *OVER if $O matches
570 JN:*STOP [jump to module *STOP if $O does not match
580 *OVER [module to begin new count (see line 560)
590 WRITE:S, [clear screen
600 WRITE:S,type run [inverse small letters writes oversize
text-color red
610 PA:60 [pause so instruction to start over can be read
620 E: [end program
630 *STOP [module to stop run
640 WRITE:S, [clear screen
650 WRITE:S,GOODBYE, [writes oversize text to
screen-color yellow
660 WRITE:S, [cleaves a blank line
670 WRITE:S,COME AGAIN! [writes oversize text to
screen-color yellow
680 E: [end run

```



...OF INTEREST

Turtle News, the newsletter of the Young Peoples LOGO Association (1208 Hillsdale Dr., Richardson, TX 75081) now includes programs for Atari PILOT, BASIC and Turtle Graphics. (Kids free, Adults \$25 year).

Closing the Gap, Microcomputers for the Handicapped, is a new publication that sounds great if this is your interest. (Route 2 Box 39, Henderson, MN 56044, \$15 year).

Mosaic (POB 708, Oregon City, OR 97045) has two new products. The Mosaic Adapter (\$60) allows you to take two of your 16K memory boards and make it one 32K board to free up the third slot, and the Expander (\$120) allows you to take one 16K board and expand it to a 32K board-especially useful in upgrading a 16K Atari 400 so you don't need to throw away your original memory board.

Synapse (820 Coventry Road, Kensington, CA 94707) has announced their upgraded FileManager 800 with Math functions and compatability with Axlon's 128K RamDisk memory board. We have ordered a RamDisk and plan on reviewing it for the next issue. We have also received the new, very powerful DataPerfect from LJK and it is under review. If we get the new FileManager in time, we will review both!!

The University of Oregon Education Department is having its 3rd Annual Summer Conference in Eugene, The Computer: An Extension of the Human Mind (\$95). Details write: '82 Summer Conference, Jude Ridge, U. of O., Eugene, OR 97402 (503)686-3405. If you come, give us a ring!

-Mike Dunn

PROGRAM PROTECTOR

BY JOHN WILEY
MICROBITS
434 W 1ST
ALBANY,OR 97321
(503) 967-9075

Suppose that you have just developed some formula that will allow you to do fantastic games in BASIC. You want to be able to market your games, but do not want the competition to find out your secret formula. What do you do?

The easiest way to make your program hard to read is to change all the variables to the same name. This is simple to do because all the variable names are stored in a variable name table (VNT) in RAM. All we have to do is change what is in the table and the next time the program is listed, all the variables are changed. The starting address of the table is pointed to by VNTP (130,131 Decimal) and the end of the table is pointed to by VNTE (132,133 Decimal). What I did was change every character in the VNT to a linefeed character. Then when the program is listed, it only does a linefeed every time it used to print the variable name.

The second part of the program is to demonstrate another feature of how BASIC works. This feature deletes BASIC statements with linenumbers larger than 32512. If our variable changer program is numbered larger than 32512 then it will delete itself from memory when done. This way, only the protected program is left and all you have to do is SAVE it.

The beginning of your program in RAM is pointed to by STMTAB (136,137 Decimal). The first two bytes in a statement are its linenumber (low byte first, high byte second; linenumber= 256* 2nd Byte + 1st Byte). The byte after the linenumber tells how many bytes long that statement is in memory. By adding this number to the current position, we are now pointing at the next statement linenumber. My routine checks through the program in memory until it finds a linenumber larger than 32512 (high byte= 127). It then changes the 127 to a 128, which makes that linenumber greater than 32768 (128*256). When BASIC gets to a linenumber larger than 32768 it assumes that that is the end of the program. Now, when the program is LISTed or RUN, BASIC doesn't know about the statements that resided above 32512; effectively deleting that portion of the program.

Lines 32600-32640 Change the variable table.
Lines 32700-32730 Delete this routine.

To use this routine, I would type it in and LIST it out on tape or disk. Then when you have the program to protect in memory, all you do is ENTER this routine into memory along with the program and type GOTO 32600. Now you just SAVE or LIST your to tape or disk and it can be LOADED or ENTERed into the computer.

```

32600 START=PEEK(130)+256*PEEK(131)
32610 VEND=PEEK(132)+256*PEEK(133)
32620 FOR X=START TO VEND
32630 POKE X,155
32640 NEXT X
32700 X=PEEK(136)+256*PEEK(137)
32710 X=X+1
32720 IF PEEK(X)>126 THEN POKE X,128:END
32730 X=X+PEEK(X+1):GOTO 32720

```

June Meeting

Meetings are always on the 2nd Weds night at 7:30. The next meeting will be at the Northwest Natural Gas Building on Goodpasture Island Road next to K-Mart. It is off Delta Highway one Exit from Eugene past Valley River Center. See the Axlon 128K RAMDISK in action, Bill Wilkinson of O.S.S. new book on the Atari DOS (we bought several copies for sale), and anything else that is new.

EPROM PROGRAMMERS

by
Ed Chastain

This article discusses several aspects which should be considered in purchasing, building, and/or using EPROM programmers with the ATARI Personal Computer System (PCS). An EPROM (Erasable programmable Read Only Memory) is a special memory device which retains the information (data or programs) stored in it. It is similar to the ROM's (Read Only Memory) ATARI uses in their game and other cartridge programs. An EPROM has an advantage over a ROM: It can be erased and reprogrammed, if necessary.

Why are we interested in them? Well, like game cartridges they are very easy to use with the ATARI PCS. After it is programmed and placed in a cartridge, just plug it in, and turn the power on. Also, for dedicated tasks and other specialized applications they are very useful. For example, if you are interested in microcomputers as a hobby. In building microprocessor based systems for home use, you may require the use of EPROM's to control the microprocessor. There are many home applications for which microprocessor based systems can be used, such as: burglar/fire alarms, temperature control, lighting, watering lawns and plants, feeding the fish, and probably anything else you can think of.

There are two basic types of EPROM programmers: the "stand alone" or "intelligent" version; and the "unintelligent" version. The intelligent version has a built in microprocessor and is capable of programming EPROM's on its own. The unintelligent version requires control from an external device, such as a microcomputer. The unintelligent version is the topic of this article. This second type offers the advantage of using the microcomputer to save the information until it's in a final form to be stored in the EPROM.

There are several other important features which should be considered in selecting an EPROM programmer. First, how does it connect to the ATARI PCS? There are a variety of ways to connect an EPROM programmer to the ATARI PCS, such as: the front jacks; RS-232 via the interface module; serial I/O port; RAM slot; and internal connections inside the ATARI case. Which of these is best? It depends upon your particular needs and how your system is arranged.

These five connection possibilities have a variety of advantages and accompanying disadvantages. Currently, there are no EPROM programmers available using all of these techniques. Perhaps in the near future all will be available. However, I do not recommend anyone with little knowledge of digital electronics purchase a kit which requires the purchaser to make connections inside the case of the ATARI PCS. This not only voids your ATARI warranty, but you may seriously damage your ATARI PCS.

The simplest EPROM programmer to build or operate is probably one which utilizes the front jacks of the ATARI's PIA (Peripheral Interface Adapter) chip. So, if the EPROM programmer you are looking at requires 16 or less I/O lines, then this is a good connection technique.

After you have determined whether or not a given EPROM programmer will work with the ATARI, next find out what it will cost. The prices I've seen run from \$25 (for a bare circuit board, no parts) to several hundred dollars for a factory assembled and tested one. However, I do expect to see the prices fall as new manufacturers enter the market.

Another important question is, what do you get for your dollars? Does the EPROM programmer you are considering come with everything you need, or are you required to purchase additional hardware or software? Often connectors, cables, and power supplies are not provided with the EPROM programmer and you may be required to spend an additional \$50 or so just to get it to work with your microcomputer. Besides the hardware aspects of getting a system going, you should also consider software requirements. Is the necessary software required to operate the EPROM programmer provided, or are you required to develop it yourself? This could be quite a task, unless you are an experienced programmer and understand what needs to be done.

Another question which should be asked is: Does the EPROM programmer come with adequate documentation? It will not be of much value if you don't know how to use it. Make sure

you have all the information you need to know, especially if you are unfamiliar with programming EPROM's.

Often you can save money if you are willing to assemble a kit, rather than purchasing an assembled one. However, it is important you have some knowledge of soldering, polarity of various components, special handling requirements for Integrated Circuits (IC's), and other considerations. If you assemble it incorrectly, you may damage some of the components of the kit. So, carefully consider your electronics expertise versus the increased cost of an assembled EPROM programmer. If you have the expertise and time, and you wish to save a few dollars, then a kit may be worth considering.

Another aspect of EPROM programmers is which type of EPROM does it program. There are 1K, 2K, 4K, and 8K byte EPROMs (K is times 1,000). You will need to determine which size is best for your application. The 2516 or 2716 (2K) EPROM is probably the most commonly used. There are several others available, such as the 2708 (1K), and 2732 (4K) EPROM.

So, before you purchase an EPROM programmer, ask yourself and the seller the following questions:

- 1) How does it connect to the ATARI PCS?
- 2) What does it cost?
- 3) Does it come with all the parts and software you need to use it with the ATARI PCS (connectors, cables, power supply, etc.)? Or, are you going to need to purchase additional hardware or software to use it?
- 4) Does it come with adequate documentation for use?
- 5) Does it come as a kit you assemble, or is it factory assembled?
- 6) Does it allow you to program only one type of EPROM? Or, can you program others, such as 2516, 2716, 2732, etc.

Ask Mr. ATARI

Dear Mr. ATARI,

I recently bought an ATARI 800. But I heard about this new graphics chip called the GTIA. How do I know if I have it, and what is this chip?

Sincerely,
Steve Fisher
San Mateo, CA

Dear Steve,

The GTIA is ATARI's new graphics chip. It allows you to access three more modes called 9, 10, and 11. There is a simple way to test for this chip. Enter the following program. If the product after you "RUN" it is a blue line, you have GTIA. If it is a red line, it is CTIA. GTIA chips are the Television Interface Adapter Chips, this one is revision G or C. GTIA chips are available from any authorized service center for a nominal charge.

```
00 REM GTIA TEST
01 REM BLUE IS GTIA
02 REM RED IS CTIA
10 GR:8
15 COLOR 1
20 SE:2,0,0
25 FOR X=50 TO 55
30 FOR Y=0 TO 300 STEP 2
40 PLOT X,Y
50 NEXT Y:NEXT X
60 END
```

Mr. ATARI
Marc Russell Benioff

Write To Mr. ATARI:

Marc Russell Benioff
50 Warmwood Way
Hillsborough, CA



**Atari Bulletin Boards
ACE BB (503)343-4352
Wash.D.C. (202)2768342
M.A.C.E (313)868-2064
Park Ave.(212)861-1212
Microbits(503)967-9075
Bulletin Board**

The ACE bulletin board has been a Beta test site for the Armudic Central Communicator. We now have the final version; it is now for sale and much improved from our original version. To see in action, call ACE in Eugene or ARMUDIC in Washington, D.C. When using it, please always use the same "handle" if you want replies. Any deviation will not flag your message for you to see. If you do not use the Board for 1 minute, you will be disconnected, so, if you download and then send to your disk drive and it takes more than 1 minute to save, you will be cut off. The program below with the use of the seperator program will allow you to download several programs and then seperate them later.

We would like to have a full time board, but need to buy a computer, etc., to do so. We may make a "Best of 1982 1/2" disk or tape as a money raiser to do so if their is enough interest. The new Anchor modem for \$100 is available with a Atari connector; we have ordered one and if it works out ok will let you know. At the present time, I have only one disk drive, so our program selection is limited to the current programs in the newsletter. I have ordered a RamDisk, and when I get that working, will be able to offer more. Our board is up most nights from about 11PM PST to 3PM. If the phone rings more than once, it is not up!!

**NOW AVAILABLE:
THE
ARMUDIC CENTRAL COMMUNICATOR: \$99.
AND THE
ARMUDIC PERSONAL COMMUNICATOR: \$79.**

The ARMUDIC Central Communicator for the ATARI Computer is more than just a Bulletin Board. The Personal Communicator is not a Bulletin board at all! In addition to in-RAM user messages, which can be read or scanned in a variety of ways-- available only on the Central Communicator--both the Central and Personal Communicator have the following features:

- A. Users can download (receive) files of arbitrary length, including BASIC programs. Make selections from the Main Menu or a separate Download Menu. Users can either terminate or temporarily interrupt download at any time.
- B. Remote users can upload (send) programs or other files to ARMUDIC. ARMUDIC automatically puts these files on the Download Menu.
- C. If the local system includes a printer, remote users can send messages of arbitrary length to it.
- D. ARMUDIC maintains a user log, on either the disk or printer, recording each remote user and each Menu choice.
- E. ARMUDIC keeps the time and date on an internal real-time clock. After being set manually by SYSOP or automatically by an external hardware clock time and date are shown before each Main Menu display and are recorded on each user-to-user and SYSOP message and on each log entry.
- F. Any Main Menu option or Download Menu file can be restricted to those who have entered either the SYSOP- or club-level password. A remote user uploading a file can specify the level of availability desired for that file.
- G. Simple DATA statements control the title and Main Menu. A Main Menu option to transmit an information file can be added by entering one DATA statement line. This line contains everything needed by ARMUDIC for listing and control. A Download Menu file is made available merely by placing it on the correct disk with the correct file name extender.

FLASH!
New Bulletin Board System
for ATARI owners,
with lots of public domain software.
(408) 942-6975
6.00 pm to 8.00 am and all day weekends

H. SYSOP may exercise ARMUDIC locally as though the local system were a remote terminal. During normal operation, transmissions by remote users are locally displayed on the ATARI screen, allowing SYSOP to monitor system use.

I. ARMUDIC sends a "good-by" message and hangs up after waiting one minute with no response from a remote user. SYSOP local use, however, has no "time-out".

J. SYSOP can "talk" to a remote user with the local keyboard and screen while the user is connected to ARMUDIC. SYSOP can also sign a remote user off at the end of the current Menu selection.

Requires a 40K 800 or 400, at least one disk drive, an 850 interface and the Hayes Smartmodem. See ARMUDIC in action! call (202) ARMUDIC (202-276-8342).

To order: check or money order to:
Frank L. Huband
1206 N. Stafford St.
Arlington, VA 22201

Brian's Arcade

Last month was pretty slow. I was supposed to receive Frogger from Ken Williams of On-Line as well as P.C.H. and Canyon Climber from Dennis Wallin of Data-Soft Inc. but I didn't. I did get an early copy of Marc Benioff's new game The Nightmare, an arcade game from John Bell called Zardon and at the S.F. Faire I got a brand new joystick called The Video Command joystick from Zicron International who makes Apple controllers.

Marc Benioff is not selling his games through John Bell at Crystalware any more; he is now selling his games through Automated Simulations (Epyx). Marc has also changed the names and improved his his earlier games. Forgotten Island has been changed to Vulcan Island, Quest for Power has been changed to King Arthur's Heir and The Crypt has been changed to The Crypt of the Undead. These games are all available from Epyx or Automated Simulations, and will be available in late July.

Zardon is a new arcade game from U.F.O. Software (17120 Monterey St. Morgan Hill, CA 95037). UFO Software is a new company formed by John and Patty Bell and features games by them. They now have two new games out: Zardon and the Haunted Palace. In Zardon you control a spaceship and you are trying to destroy the emerald in the heart of the Zardonian empire. The empire is guarded by very smart androids, highly explosive mines and extremely difficult barriers which you must move around to destroy the emerald. Zardon uses very good horizontal scrolling and pretty good sound effects. The game I think costs \$29.95 and is available from UFO. Haunted Palace is a new adventure game which I think is available from UFO software for \$29.95 also. The Haunted Palace uses extraordinary 3-D graphics which are the first I have seen in a game for the Atari computer. The reason I'm not sure if it is available from UFO or not is because when I received it from John Bell UFO was not around and the package said Crystalware.

That's all for this month. Next month I hope to have Canyon Climber and PCH from Data-Soft, Frogger from On-Line and I'll also be reviewing the new joystick that I mentioned in the first paragraph.

-Brian Dunn

BANKSHOT

by Stan Ockers, Lockport, IL

Here it is ... yet one more attempt at a Player Missile routine which makes player movement 'simple'. The problem with the 'universal' routine in 'Chicken' was it was almost completely independent of Basic. Once set up, everything continued to move, even if the Basic program was stopped. Other problems seemed to crop up because a separate load player routine was used. If a vertical blank interrupt occurred part way through the loading process, unexpected things could happen. The load routine should be part of the VBI movement routine.

I decided to start from scratch. This time I've provided an assembly listing of what I came up with. As usual there are a number of special locations in page six. Each player has locations which can be POKEd with X and Y positions as well as one which can be POKEd with a number which determines the image used. Table 1 lists page six locations. The routine itself is completely relocatable and is stored in VB\$.

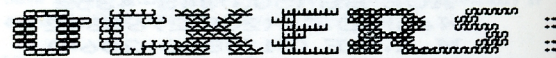
All players are updated during each VBI ... horizontal positions are always updated while vertical positions are updated only if the new vertical position for that player is different from the old. In such cases, zeros are written into the appropriate player area until the proper vertical position is reached. The image to insert is determined by an even number 0,2,4,... The numbers refer to a set of pointers loaded into page six starting at 1568. The pointers are the addresses of strings which hold the images. After loading the image, the rest of the area for that player is zeroed out. Jumps from one point on the screen to a second point quite far away can be immediate. To show motion the intervening positions are used.

What has to be set up for this routine to work? Well, first of all the routine is only for single line resolution and only for 4 players (no missiles). Refer to the listing of 'Bankshot' for the lines which follow. The routine has to be put into VB\$ (lines 250-302) and its address put into a machine language insertion routine (line 380, this routine hooks our routine into VBI). Room has to be set aside for the player-missiles and the Hi byte of the address of this area given to 1551 as well as CTIA (line 610). All strings representing images have to be filled and their addresses (Lo and Hi bytes) placed in the previously mentioned set of pointers (lines 624-645). Image 0 (the first in the list) should probably be the single character zero. This can be used to clear a player area although setting the horizontal position to 0 will get the image off the screen as well. Initial horizontal and vertical positions should be POKEd as in lines 650-660. The old and new vertical positions should be different or the images will never be loaded until they are. The image numbers for the different players have to be POKEd into locations 1552-1555 (line 690) and the normal PM single line resolution initialization given (line 670). Colors can be set (line 695) and finally, the VBI routine inserted (line 710).

The game 'Bankshot' is fairly straight forward. There is one problem I was unable to fix. You are not supposed to shoot directly at the pockets. I put a flag (BOUNCE = 1 or 0) to determine if you do, but it doesn't always work. This is apparently due to the size of the ball and the configuration of the corner pockets. It's easy to have a collision with the rail almost simultaneously with the ball touching the pocket. I've tried extending the corner pockets out but that makes it too easy to sink the shot. If anyone figures out a way to handle this problem, let me know.

1536-1545 (0600-0609) Insertion routine
1551 (060F) PM Area Hi byte
1552-1555 (0610-0603) Image #'s (even only)
1556-1559 (0614-0617) Plyr horiz. pos.
1560-1563 (0618-061B) Old vert. pos.
1564-1567 (061C-061F) New vert. pos.
1568 and up (0620) Pointers to images (Lo and Hi bytes)

Note: 00CB-00CF also used Table 1.



Assembly language routines for BANKSHOT

by Stan Ockers

```
0614      10 HPOS      = $0614
0618      20 OLDV      = $0618
061C      30 NEWV      = $061C
00CB      40 APTR      = $00CB
00CD      50 IPTR      = $00CD
060F      60 PMPG      = $060F
0610      70 INGN      = $0610
0620      80 INGP      = $0620
00CF      90 STOR      = $00CF
D000      0100 PLYRH   = $D000
0000      0110      X= $0000
0000 18      0120 VBI   CLC
0001 A00F06 0130      LDA PMPG      ADD TO PM HI ...
0004 6904    0140      ADC #04      FOUR PAGES TO GET ...
0006 85CC     0150      STA APTR+1  PLYR AREA PTR HI
0008 A200     0160      LDX #00      X IS PLYR COUNTER
000A 86CF     0170      STX STOR
000C A800     0180      LDY #00      Y USED IN INDIR OPER.
000E 84CB     0190      STY APTR
0010 B01406 0200 HORIZ LDA HPOS,X   UPDATE HOR. POS.
0013 9000D0 0210      STA PLYRH,X
0016 B01C06 0220      LDA NEWV,X   CHECK IF VERT. POS.
0019 D01806 0230      CMP OLDV,X   CHANGED
001C F03F     0240      BEQ NXTPLYR SKIP IF NOT
001E B01C06 0250      LDA NEWV,X   UPDATE OLD VERT.
0021 9D1806 0260      STA OLDV,X   POSITION
0024 A5CB     0270 ZERO1 LDA APTR   CK. IF REACHED ..
0026 D01C06 0280      CMP NEWV,X   VERTICAL POS. YET
0029 F00A     0290      BEQ LDPLYR  IF SO, START LOAD
002B A900     0300      LDA #00      ELSE FILL WITH ...
002D 91CB     0310      STA (APTR),Y ZEROS...
002F E6CB     0320      INC APTR    UNTIL YOU DO
0031 F02A     0330      BEQ NXTPLYR (OUT OF PLYR AREA)
0033 D0EF     0340      BNE ZERO1
0035 B01006 0350 LDPLYR LDA INGN,X   USE IMAGE NUMBER ...
0038 AA       0360      TAX          AS INDEX AND ...
0039 B02006 0370      LDA INGP,X   SET UP INDIR.
003C 85CD     0380      STA IPTR    POINTER
003E B02106 0390      LDA INGP+1,X
0041 85CE     0400      STA IPTR+1
0043 B1CD     0410 PLBYTE LDA (IPTR),Y GET IMAGE...
0045 F00E     0420      BEQ CLREST  IF A ZERO, FINISHED
0047 91CB     0430      STA (APTR),Y PUT IN PLYR AREA
0049 E6CD     0440      INC IPTR    NEXT IMAGE BYTE
004B D002     0450      BNE UPONE   NO PAGE CROSSING
004D E6CE     0460      INC IPTR+1  PAGE CROSSING
004F E6CB     0470 UPONE INC APTR    NEXT AREA BYTE
0051 F00A     0480      BEQ NXTPLYR REACHED END OF AREA
0053 D0EE     0490      BNE PLBYTE NEXT IMAGE BYTE
0055 A900     0500 CLREST LDA #00    ZERO OUT REST ..
0057 91CB     0510 ZERO2 STA (APTR),Y OF PLYR AREA
0059 E6CB     0520      INC APTR
005B D0FA     0530      BNE ZERO2
005D E6CC     0540 NXTPLYR INC APTR+1 NEXT AREA
005F A6CF     0550      LDX STOR    RECOVER PLYR INDEX
0061 E8       0560      INX        NEXT PLYR
0062 86CF     0570      STX STOR    SAVE INDEX
0064 E004     0580      CPX #04     LAST PLAYER?
0066 D0AB     0590      BNE HORIZ   NO, GET ANOTHER
0068 4C62E4 0600      JMP #E462    BACK TO ATARI'S VBI
0068         0610      .END
```

```
=0614 HPOS      =0618 OLDV      =061C NEWV      =00CB APTR
=00CD IPTR      =060F PMPG      =0610 INGN      =0620 INGP
=00CF STOR      =D000 PLYRH     0000 VBI        0010 HORIZ
005D NXTPLYR    0024 ZERO1     0035 LDPLYR     0043 PLBYTE
0055 CLREST     004F UPONE     0057 ZERO2
```


BANKSHOT

```

0 REM *****
1 REM * ACE NEWSLETTER *
2 REM * JUNE 1982 *
3 REM *3662 VINE MAPLE *
4 REM *EUGENE, OR 97405*
5 REM * $10 YEAR *
6 REM *****
10 REM *****
20 REM ** BANKSHOT **
30 REM ** S.O. 4/82 **
40 REM *****
90 OPEN #1,4,0,"K":GRAPHICS 1
8:POSITION 6,3: ? #6:"BankShot"
:REM INVERSE An and oT
100 POSITION 4,6: ? #6: ? # PLAYE
RS?"DIM B$(16)
110 GET #1,NP:NP=NP-48:IF NP<1
OR NP>9 THEN 110
120 DIM SCORE(NP):POSITION 15,
6: ? #6:NP:POSITION 4,9: ? #6:"i
nstructions?" :GET #1,A:IF A=89
THEN GOSUB 2000
250 REM * VBI ROUTINE UPDATED
BY POKES *
260 DIM VB$(107):FOR I=1 TO 10
7:READ A:VB$(I,D)=CHR$(A):NEXT
I
270 DATA 24,173,15,6,105,4,133
,204,162,0,134,207,160,0,132,2
03,189,20,6,157,0,208,189,28,6
,221,24,6
280 DATA 240,63,189,28,6,157,2
4,6,165,203,221,28,6,240,10,16
9,0,145,203,230,203,240,42
290 DATA 208,239,182,16,6,170,
189,32,6,133,205,189,33,6,133,
206,177,205,240,14,145,203,230
,205,208,2
300 DATA 230,206,230,203,240,1
0,208,238,169,0,145,203,230,20
3,208,250,230,204,166,207,232,
134,207,224,4,208,168
302 DATA 76,98,228
350 REM * PAGE 6 - INSERT VBI
ROUTINE *
360 FOR I=1536 TO 1545:READ A:
POKE I,A:NEXT I
370 DATA 104,160,0,162,0,169,7
,76,92,228
380 A=ADR(VB$):B=INT(A/256):C=
A-256*B:POKE 1538,C:POKE 1540,
B
400 GRAPHICS 3:COLOR 1:PLOT 3,
0:DRAWTO 36,0:DRAWTO 36,19:DRA
WTO 3,19:DRAWTO 3,0
410 POKE 708,204:POKE 709,0:PO
KE 712,233:COLOR 2:FOR I=1 TO
14:READ X,Y:PLOT X,Y:NEXT I
420 DATA 3,0,3,1,4,0,19,0,36,0
,36,1,35,0,35,19,36,19,36,18,1
9,19,3,19,3,18,4,19
430 A=PEEK(560)+256*PEEK(561)
440 IF PEEK(A)>66 THEN A=A+1:
GOTO 440
450 POKE A,71:POKE A+3,6:POKE
A+4,6:POKE A+5,65:POKE A+6,PEE
K(A+7):POKE A+7,PEEK(A+8)
480 POKE 656,0:POKE 657,24: ? "
Ball Player":GOSUB 1500:REM
INVERSE BALL PLYER
600 REM * PLAYER MISSILE SETUP
*
610 A=PEEK(106)-16:POKE 54279,
A:POKE 1551,A

```

```

620 REM ** IMAGE 4 **
624 DIM CUR$(6):FOR I=1 TO 6:R
EAD A:CUR$(I,I)=CHR$(A):NEXT I
:CUR=ADR(CUR$):POKE 1573,INT(C
UR/256)
625 POKE 1572,CUR-256*PEEK(157
3)
626 DATA 16,16,56,16,16,0
628 REM ** IMAGE 2 **
630 DIM BALL$(9):FOR I=1 TO 9:
READ A:BALL$(I,I)=CHR$(A):NEXT
I:BAL=ADR(BALL$):POKE 1571,IN
T(BAL/256)
632 POKE 1570,BAL-256*PEEK(157
1)
640 DATA 16,56,108,116,124,124
,56,16,0
642 REM ** IMAGE 0 **
645 DIM Z$(1):Z$=CHR$(0):ZERO=
ADR(Z$):POKE 1569,INT(ZERO/256
):POKE 1568,ZERO-256*PEEK(1569
)
650 FOR I=1556 TO 1567:READ A:
POKE I,A:NEXT I
660 DATA 120,100,86,86,100,30,
53,53,120,40,63,63
670 POKE 559,62:POKE 53277,3
690 POKE 1552,2:POKE 1553,4:PO
KE 1554,0:POKE 1555,0
695 POKE 704,36:POKE 705,0
700 A=USR(1536)
710 YMIN=38:YMAX=177:XMIN=62:X
MAX=186:BALL=0:PLYR=0:B$="OOOO
OOOOOOOOOOO":GOSUB 1500:REM I
NVERSE O'S
720 FOR I=1 TO NP:SCORE(I)=0:IN
EXT I
730 POKE 77,0:BALL=BALL+1:IF B
ALL>15 THEN GOTO 3000
735 POKE 656,0:POKE 657,5: ? BA
LL
740 X=INT(RND(0)*100+70):Y=INT
(RND(0)*100+70):POKE 1556,X:PO
KE 1564,Y
743 IF FL=1 THEN FL=0:GOTO 750
745 PLYR=PLYR+1:IF PLYR>NP THE
N PLYR=1
747 POKE 656,0:POKE 657,14: ? P
LYR
749 REM ** MOVE CURSOR ALONG R
AIL **
750 S=STICK(0):XP=PEEK(1557):Y
P=PEEK(1565):X=PEEK(1556):Y=PE
EK(1560)
760 IF XP>XMIN AND XP<XMAX THE
N 780
765 IF (S=10 OR S=14 OR S=6) A
ND YP>YMIN+2 THEN YP=YP-1
770 IF (S=9 OR S=13 OR S=5) AN
D YP<YMAX THEN YP=YP+1
780 IF YP>YMIN+2 AND YP<YMAX T
HEN 795
785 IF (S=5 OR S=6 OR S=7) AND
XP<XMAX THEN XP=XP+1
790 IF (S=9 OR S=10 OR S=11) A
ND XP>XMIN THEN XP=XP-1
795 POKE 1557,XP:POKE 1565,YP
800 IF STRIG(0)=1 THEN 750
805 R=SQR((XP-X)*(XP-X)+(YP-Y)
*(YP-Y))
810 DELY=2*(YP-Y)/R:DELX=2*(XP
-X)/R
880 BOUNCE=0:MOVE=0
885 REM ** LOOP TO MOVE BALL *

```

6x
34
46

```

800 IF STRIG(0)=1 THEN 750
805 R=SQR((XP-X)*(XP-X)+(YP-Y)
*(YP-Y))
810 DELY=2*(YP-Y)/R:DELX=2*(XP
-X)/R
880 BOUNCE=0:MOVE=0
885 REM ** LOOP TO MOVE BALL *
*
890 X=X+DELX:Y=Y+DELY
892 IF PEEK(53252)=2 THEN GOSUB
B 1000:GOTO 940
900 IF X>XMAX OR X<XMIN+1 TH
EN DELX=-DELX:GOSUB 1800
910 IF Y>YMAX OR Y<YMIN+1 TH
EN DELY=-DELY:GOSUB 1800
920 POKE 1556,X:POKE 1564,Y
930 POKE 53278,0:MOVE=MOVE+1:IF
MOVE<150 THEN 890
940 POKE 53278,0
970 REM ** FL=1 IF SOMEBODY SC
ORED **
980 IF FL=1 THEN 730
982 REM ** FL=2 IF SHOT INTO P
OCKET **
985 IF FL=2 THEN FL=0:GOTO 740
990 GOTO 745
1000 POKE 1556,0:IF BOUNCE=0 T
HEN POKE 656,1:POKE 657,3: ? "M
UST BANK SHOT!!":FL=2:GOTO 1
020
1010 SCORE(PLYR)=SCORE(PLYR)+1
:FL=1:B$(BALL,BALL)=CHR$(48+PL
YR):GOSUB 1500
1020 FOR I=1 TO 30:SOUND 0,150
+I,12,10:NEXT I:FOR I=120 TO 1
80:SOUND 0,1,10,10:SOUND 0,0,0
,0:NEXT I
1030 FOR I=50 TO 30 STEP -5:GO
SUB 1600:NEXT I:GOSUB 1500:RET
URN
1500 POKE 656,1:POKE 657,3: ? B
$:RETURN
1600 SOUND 0,1,10,10:FOR J=1 T
O 5:NEXT J:SOUND 0,0,0,0:RETUR
N
1800 BOUNCE=1:SOUND 0,50,10,10
:FOR I=1 TO 5:NEXT I:SOUND 0,0
,0,0:RETURN
2000 ? CHR$(125):POKE 752,1:PO
SITION 8,3: ? "INSTRUCTIONS"?
: ? "This is a game of one ball
pool for"
2010 ? "up to 9 players. All
players use": ? "joystick 1. B
alls are made one at"
2020 ? "a time and must hit at
least one": ? "rail before goi
ng in a pocket to"
2030 ? "count. If a player ma
kes a ball," : ? "he gets anothe
r turn."
2040 ? : ? "Position the cursor
on the rail": ? "where you wa
nt the ball to first"
2050 ? "hit. The scorecard at
the bottom": ? "keeps track of
which player hit the"
2060 ? "balls"
2070 ? : ? : ? "Press any key to
begin the game."
2080 GET #1,A:RETURN
3000 GRAPHICS 18:POSITION 5,0:
? #6:"Final Score":FOR I=1 TO
NP:POSITION 3,I: ? #6:"Plyr #";
I: ? - "SCORE(I)
3005 NEXT I
3010 GOTO 3010

```


HOW TO USE JONESTRM

A TERMINAL PROGRAM FOR THE ATARI

program by Frank Jones
Starting Up

JONESTRM is a Download/Upload terminal program for the Atari computer. It was written by Frank Jones in BASIC and assembly language to combine ease in setting up and speed when in the actual terminal mode.

NOTE that an AUTORUN.SYS file with the RS-232-handler boot routine MUST be on disk and booted when you turn on your machine, if you are using a disk. Both DOS and NEW erase the handler, so if you ENTER JONESTRM, SAVE it to disk, reboot with AUTORUN.SYS, and then RUN the SAVED version.

JONESTRM is a BASIC program, and when it is RUN it POKes the machine language routine into a string called PROG*. During this time the screen is black. After the set up period is over, the first menu appears on the screen along with information about the size and location in RAM of the available text buffer. All menu choices are made by simply typing the appropriate key that is highlighted in inverse video. (Type an ordinary character, not an inverse video one.)

The first choice to be made is whether you wish to Download a file from the host computer or Upload a file to the host computer. If one wishes to do simple communication without file transfer Download is the proper mode to choose. After choosing between the Download and Upload modes the next choice is among no translation (None), Light translation, and ATASCII. With Light translation all high order bits are stripped from all outgoing and incoming characters and the ATASCII EOL character (155) is changed to the ASCII CR character (13) on output and vice versa on input. No translation means that the 850 Interface Module does no changing of characters during either input or output. However, be warned that the program does some translation itself; more about that later.

The next choice is between the various modes of outgoing parity setting.

(NOTE incoming parity is not checked or changed by this program.)

You should always select None if no translation has been selected because setting the parity on output will change the high order bit that was presumably to be preserved. This option was included for those users that wish to access mainframe computers that require certain parity configurations.

At this point if Upload was chosen you will be asked for the filespec of the file to be uploaded. When this has been entered the file will be loaded into RAM and then listed to the screen as a check. You will then enter the terminal mode. If Download was chosen you will go directly from the parity choice to the terminal mode without going through the file loading routine.

Terminal Operations

Whenever you enter the terminal mode the flag (ie. inverse video word) TERMINAL will appear at the top of the screen. This is to inform you that you are now in the machine language part of JONESTRM. While you are in this mode you may send and receive data from a host computer provided all of the appropriate connections have been made. You may toggle the memory save function off and on by pressing the SELECT button; the flags MEMSTORE ON and MEMSTORE OFF will be printed on the screen as you toggle the memory. While the memory save option is in effect all incoming characters will be stored in sequence until the buffer is full. If the buffer should fill up the flag MEMORY FULL will be printed on the screen.

NOTE if you have filled your buffer prior to an Upload you should NOT turn on the memory save feature until you have completed the Upload. Otherwise the incoming characters will overwrite your file.

When you enter the terminal mode you will be in full duplex ie. only those characters that are received are printed on the screen and stored in memory. If the host computer echos all characters that it receives these characters will be incoming and will be printed and saved if desired. If the host computer operates in half duplex it cannot send and receive at the same time so it will not echo the characters that it receives from you. In this case you should turn on the half duplex mode. You can toggle between half and full duplex by pressing the OPTION button. Whenever you do the flags HALF DUPLEX and FULL DUPLEX will be printed on the screen as appropriate.

Leaving Terminal Mode

When you are ready to leave the terminal mode you may do so by pressing the START button. When you do one of three things will happen depending on the circumstances. If you have chosen the Upload option and have not uploaded the file you will go into the upload mode. The flag UPLOADING will appear on your screen and the buffer will be dumped to the computer on the other end of the line. When the upload is complete you will reenter the terminal mode, this time in Download mode.

If you were in Download mode and saved anything at all in memory when you press START you will be asked for the filespec of the file to which you wish to save your memory. This can be the cassette (C:), the printer (P:), the screen editor (E:), or a disk file (D:FILENAME). After entering the filespec the saved memory will be written to the file and you will be told that you may reenter the terminal mode by pressing START. If, however, you wish to save the memory to another file before returning to the terminal mode press START and BEFORE RELEASING THE START BUTTON press the OPTION button. This will bring you back to the request for a filespec. This may be repeated as many times as wished.

If you exit the terminal mode without saving anything to memory you will automatically bail back out to the main menu and may start another session with different parameters if you wish.

Internal Translations And Other Features

When you choose among Light & No translation and ATASCII in the second menu you are choosing the configuration of your 850 Interface Module RS232 ports. You should read your 850 instruction manual for information about these configurations. This program does some additional translation of its own, however. First of all, nothing that comes in from the port is changed at all before it is stored in memory. Therefore if you choose ATASCII or No translation you will save in memory everything EXACTLY as it was sent. There will be some translation, however, before it is displayed on the screen. First of all no control characters (ASCII values < 32) are displayed. This means that you will not see line feeds for instance; they will, however be stored and can mess up a program that you are downloading. You should NOT ask for linefeeds; you DO NOT need them even if the test messages are single spaced. The carriage return character (ASCII 13) is translated to the ATASCII EOL character. The printer bell character (ASCII 7) is translated to the console bell (ATASCII 253). Finally the ASCII backspace character (ASCII 8) is changed to the ATASCII DELETE/BACKSPACE (ATASCII 126). Again, none of this translation affects what is stored in memory; everything is stored exactly as it is received.

In ATASCII mode, no characters are changed on output. In No translation mode, two characters are changed on output. The DELETE/BACKSPACE character (ATASCII 126) is changed to 8, the ASCII backspace character, so that the key will have the same function with most host computers that it does in the Atari. Also the RETURN key or EOL (ATASCII 155) is changed to the ASCII carriage return (ASCII 13) before it is sent. In light translation the 850 module would do this translation automatically but in the no translation mode it would not be done. I felt that there were enough situations in which inverse video characters (ASCII values >= 128) could be sent and received but the host computer would still not recognize the EOL character to warrant this feature.

In half duplex operation after a character has been sent to the port it is then handed over to the input routine and handled just like any other incoming character.

An additional JONESTRM feature is the ability to send a computer "BREAK" by simply pressing the BREAK key. This will cause the screen to flash, a beep to sound, the flag BREAK to be printed on the screen, and last but not least a true break signal (approx. 0.5 sec. of SPACE tone) to be sent.

The memory save feature will be turned off and the duplex will be set at full and should be reset to the desired setting. Sending the BREAK signal will not be of much use when connected to a BBS since most of them do not recognize it but it can be essential when you are connected to a mainframe computer whose attention cannot be gotten any other way.

That about sums up the facts about JONESTRM. Enjoy it.

```

5 REM UPLOAD/DOWNLOAD TERMINAL
VER.2.3 by Frank C. Jones May
3,1982
10 DIM PROG$(379),PROG2$(7),SP
OOL$(15)
20 CON=53279:POKE 559,0:IF PEE
K(ADR(PROG$))=104 THEN 40
30 FOR I=1 TO 379:READ A:PROG$
(I,I)=CHR$(A):NEXT I
40 DIM MSG$(65):RESTORE 1000:IF
OR I=1 TO 65:READ A:MSG$(I,I)=
CHR$(A):NEXT I
50 DIM S$(5),T$(8),U$(9):FOR I
=1 TO 5:READ A:S$(I,I)=CHR$(A)
:NEXT I:FOR I=1 TO 8:READ A:T$
(I,I)=CHR$(A):NEXT I
60 FOR I=1 TO 9:READ A:U$(I,I)
=CHR$(A):NEXT I:DIM BR$(7):FOR
I=1 TO 7:READ A:BR$(I,I)=CHR$
(A):NEXT I
65 FOR I=1 TO 7:READ A:PROG2$(
I,I)=CHR$(A):NEXT I
70 N=PRE(0)-256:N=N*(N<=32767)
+32767*(ND32767):DIM TXT$(N)
80 SETCOLOR 2,9,0:PROG$(200,20
0)=CHR$(13):PROG$(192,192)=CHR
$(8)
90 POKE 82,0:PRINT " ";
100 PRINT N-1;" BYTES OF MEMOR
Y AVAILABLE:"PRINT "FROM-":ADR
(TXT$):" TO-":ADR(TXT$)+N-2
110 CLOSE #1:OPEN #1,4,0,"K"
120 POKE 752,1:PRINT "Opera
tion Mode:"PRINT :PRINT "";C
HR$(196):"ownload":PRINT :PRIN
T "";CHR$(213):"pload"
130 POKE 559,34:POKE 752,0:GET
#1,ANS:IF ANS=68 THEN UPDL=0:
GOTO 160
140 IF ANS=85 THEN UPDL=1:GOTO
160
150 GOTO 90
160 POKE 752,1:PRINT " )Tran
slation Mode:"PRINT :PRINT "
";CHR$(206):"one":PRINT :PRINT
"";CHR$(204):"ight"
165 PRINT :PRINT "";CHR$(193)
:"TASCII"
170 POKE 752,0:GET #1,ANS:IF A
NS=76 THEN MODE=0:GOTO 200
180 IF ANS=78 THEN MODE=32:GOT
O 200
185 IF ANS=65 THEN MODE=0:PROG
$(200,200)=CHR$(155):PROG$(192
,192)=CHR$(126):GOTO 200
190 GOTO 160
200 POKE 752,1:PRINT " )Pari
ty:"PRINT :PRINT "";CHR$(206)
:"one":PRINT :PRINT "";CHR$(
207):"dd"
210 PRINT :PRINT "";CHR$(197)
:"ven":PRINT :PRINT "";CHR$(2
11):"et"
220 POKE 752,0:GET #1,A:IF A=7
8 THEN PARITY=0:GOTO 270
230 IF A=79 THEN PARITY=1:GOTO
270
240 IF A=69 THEN PARITY=2:GOTO
270
250 IF A=83 THEN PARITY=3:GOTO
270
260 GOTO 200
270 IF UPDL THEN GOSUB 410

```

```

280 PRINT " )T$":POKE 65,0:A
=ADR(TXT$)
290 CLOSE #2:OPEN #2,13,0,"R":
XIO 38,#2,MODE=PARITY,0,"R":XI
O 40,#2,0,0,"R"
300 A=USR(ADR(PROG$),A,ADR(TXT
$)+N-1,ADRMMSG$):IF PEEK(207)
=128 THEN 520
305 IF A=ADR(TXT$) AND NOT UP
LD THEN CLOSE #2:GOTO 80
310 ON UPDL+1 GOSUB 380,480
320 IF UPDL THEN UPDL=0:TXT$=""
:GOTO 280
330 PRINT "PRESS ";S$;" TO RE-
ENTER TERMINAL MODE"
340 IF PEEK(CON)>6 THEN 340
350 IF PEEK(CON)=6 THEN 350
360 IF PEEK(CON)=2 THEN 310
370 GOTO 280
380 CLOSE #2:PRINT " )ENTE
R OUTPUT FILENAME:"PRINT :PRIN
T "";POKE 702,64:POKE 65,3
390 TRAP 490:INPUT SPOOL$:CLOS
E #3:OPEN #3,8,0,SPOOL$:IF SPO
OL$(1,1)="E" THEN SETCOLOR 2,9
,0
400 TXT$(A-ADR(TXT$)+1)=" " :PR
INT #3:TXT$:CLOSE #3:RETURN
410 PRINT " )ENTER UPLOAD
FILENAME:"PRINT :PRINT "";PO
KE 702,64:INPUT SPOOL$:TXT$=""
420 TRAP 490:CLOSE #3:OPEN #3,
4,0,SPOOL$:TRAP 40000:POKE 65,
3
430 AD=ADR(TXT$):XX=INT(AD/256
):WW=AD-XX*256:ZZ=INT((N-1)/25
6):YY=(N-1)-ZZ*256
440 GOSUB 540:TXT$(QQ+1)=" "
450 IF PEEK(883)=136 THEN 470
460 PRINT "ERROR ";PEEK(883):"
DURING TEXT LOAD":STOP
470 CLOSE #3:PRINT TXT$:RETURN

480 PRINT " )U$":PRINT #
2:TXT$:RETURN
490 PRINT " )UNABLE TO OPE
N ";SPOOL$:PRINT "PRESS ";S$;
" WHEN READY"
500 IF PEEK(CON)>6 THEN 500
510 GOTO PEEK(186)+256*PEEK(18
7)-10
520 SETCOLOR 2,13,10:SOUND 0,3
0,10,15:XIO 34,#2,2,15,"R":FOR
I=1 TO 20:NEXT I:XIO 34,#2,3,
0,"R"
525 SOUND 0,0,0,0:SETCOLOR 2,9
,0
530 PRINT BR$:GOTO 290
540 POKE 882,7:POKE 884,WW:POK
E 885,XX:POKE 888,YY:POKE 889,
ZZ
550 K=USR(ADR(PROG2$),48)
560 QQ=PEEK(888)+256*PEEK(889)
:RETURN
600 DATA 104,104,133,213,104,1
33,212,104,133,215,104,133,214
,104,133,225,104,133,224,169,1
28,133,216,169,0
610 DATA 133,226,133,207,172,3
1,208,192,7,240,112,192,6,208,
1,96,192,5,208,32,172,31,208,1
92,5

```

```

620 DATA 240,249,164,216,192,2
55,240,90,152,73,128,133,216,2
08,6,169,12,133,217,208,36,169
,25,133,217
630 DATA 208,30,192,3,208,67,1
72,31,208,192,3,240,249,164,22
6,152,73,128,133,226,208,6,169
,51,133
640 DATA 217,208,4,169,38,133,
217,24,165,224,101,217,141,68,
3,165,225,105,0,141,69,3,169,1
4,141
650 DATA 72,3,169,0,141,73,3,1
69,11,141,66,3,162,0,32,86,228
,169,0,240,2,240,137,173,252
660 DATA 2,201,255,240,54,162,
32,169,11,157,66,3,169,0,157,7
2,3,157,73,3,162,16,157,72,3
670 DATA 157,73,3,169,7,157,66
,3,32,86,228,201,126,208,4,169
,8,208,6,201,155,208,2,169,13
680 DATA 162,32,32,86,228,164,
226,208,50,165,17,208,9,169,12
8,133,17,133,207,96,240,243,16
2,32,169
690 DATA 13,157,66,3,32,86,228
,173,235,2,201,0,240,163,169,7
,157,66,3,169,0,157,72,3,157
700 DATA 73,3,32,86,228,192,15
4,240,210,164,216,208,10,162,0
,129,212,230,212,208,2,230,213
,201,13
710 DATA 208,4,169,155,208,20,
201,7,208,4,169,253,208,12,201
,8,208,4,169,126,208,4,201,32,
144
720 DATA 18,160,11,140,66,3,16
0,0,140,72,3,140,73,3,162,0,32
,86,228,165,215,197,213,144,16
730 DATA 240,2,208,136,165,214
,197,212,144,6,240,4,169,0,240
,135,169,255,133,216,165,224,1
41,68,3
740 DATA 165,225,141,69,3,169,
13,141,72,3,169,0,141,73,3,169
,11,141,66,3,162,0,32,86,228
750 DATA 169,0,240,216
1000 DATA 155,205,197,205,207,
210,217,160,198,213,204,204,15
5,205,197,205,211,212,207,210
1010 DATA 197,160,207,206,160,
155,205,197,205,211
1020 DATA 212,207,210,197,160,
207,198,198,155,200,193,204,19
8,160,196,213,208,204,197,216,
160,155
1030 DATA 198,213,204,204,160,
196,213,208,204,197,216,160,15
5
1040 DATA 211,212,193,210,212
1050 DATA 212,197,210,205,201,
206,193,204
1060 DATA 213,208,204,207,193,
196,201,206,199,155,194,210,19
7,193,203,155, 104,104,104,17
0,76,86,228

```


You can download Separator from
from our Bulletin Board.

```
1 REM PROGRAM SEPARATOR by Frank C. Jones
2 REM With this program you can download several programs at a single session and then separate them
3 REM before ENTERing them, First store the entire download to a file, then LOAD and RUN this program.
4 REM It will ask for a string to use as a separator. It will then search out each occurrence of the chosen
5 REM string (up to the maximum number you have given it) and mark the positions as delimiters of the
6 REM separate programs. The word "Option" would be a good choice as a separator string; it always follows
7 REM an individual download and lower case letters are not found too often in programs.
10 PRINT ">HOW MANY PROGRAMS (MAX.) ARE IN THE FILE?";
PRINT "";
```

```
20 INPUT N
30 DIM DELIM(N),AN$(1),IN$(120)
40 DIM S$(120):PRINT ">ENTER DELIMITER STRING:";PRINT ""
50 DIM FILE$(120):PRINT ">ENTER INPUT FILENAME:";PRINT ""
60 S=FRE(0)-256:MAX=32767:S=S*(S<=MAX)+MAX*(S>MAX)
70 DIM TXT$(S)
80 CLOSE #1:OPEN #1,4,0,FILE$
90 TRAP 500:INPUT #1,IN$:IN$(LEN(IN$)+1)=CHR$(155):TXT$(LEN(TXT$)+1)=IN$:GOTO 90
95 D=0:L=LEN(S):L1=LEN(TXT$):PRINT ">SEARCHING"
100 FOR I=1 TO L1-L+1
110 IF TXT$(L1-L+1)=S$ THEN DELIM(D)=D+1:IF D>N+1 THEN GOSUB 600:POP #D=N+1:GOTO 120
115 NEXT I
120 PRINT ">THE STRING ""S$"" HAS BEEN"";PRINT "FOUND"";D"" TIMES"
130 PRINT ">ENTER PROGRAM # (0 TO EXIT):";PRINT "OUTPUT FILENAME:";PRINT "";INPUT NO,FILE$
```

```
140 IF NO=0 THEN END
150 IF NO=D+1 THEN GOSUB 210
160 I=1:IF NO<>1 THEN I=DELIM(NO-2)
165 J=LEN(TXT$):IF NO<=N+1 THEN J=DELIM(NO-1)
170 CLOSE #1:OPEN #1,8,0,FILE$
180 PRINT #1;TXT$(I,J)
190 CLOSE #1
200 GOTO 120
210 FOR I=DELIM(NO-2) TO LEN(TXT$):AN$=TXT$(I,J):PRINT AN$;NEXT I
220 PRINT "DO YOU WISH TO SAVE THIS?";INPUT AN$:IF AN$="Y" THEN RETURN
230 POP:GOTO 120
500 IF PEEK(195)=136 THEN 95
510 PRINT ">"";PRINT ">ERROR -"";PEEK(195);"" ON FILE ENTRY"";END
600 PRINT ">THE STRING ""S$"""";PRINT "HAS BEEN FOUND"";PRINT "MORE THAN ""N;"" TIMES S."
610 PRINT "IGNORING THE REST."";FOR I=1 TO 1000:NEXT I:RETURN
```

RANDOM ACCESS PROCESSING

By Kirt E. Stockwell

PART 3

In order to minimize the time span that this series of articles will run, we must make some assumptions about your understanding of Atari string handling. If you have been reading the series on Random Access and the series on Structure in Basic in this newsletter, you should be able to follow this. If not, string along with us and maybe we can help you see the light.

There are 4 (four) basic functions or activities carried on with respect to random access data files. These include building the initial file, adding records, reading specific records, and updating specific records. Some crazies even go so far as to MERGE data files, but, since this has nothing to do with random access, we won't get into it. The process of building the initial file is the least complex of the operations. This requires only that the records be built individually, then recorded on disk, using the NOTE function and saving the location on an appropriate KEY file. Since the records are initially entered sequentially, there is not much difference here between building a sequential access and a random access file. Keep in mind that Random Access files can be read sequentially. The only difference between a random access file and a sequential file on the Atari is that you have kept track of where the records are located in the random access file. (Just to get in trouble, we should mention here that the method we have chosen for this series of articles is only one of several.) There are methods of building Random Access files that render them useless for sequential access. We won't bother with those.

It must be remembered that the Key file is crucial to the use of the Random access file. Therefore, when we wish to read a specific record, we must search for its location by scanning the key file. Then, we POINT to the sector and byte at which the record begins, and read the record. The same thing must be done to UPDATE a given record. Read it, keeping track of the starting location, modify it as needed, and POINT to sector and byte that was saved when you read the record. When you write the record back out, it will then be placed in the same location it was in previously.

To ADD a record, open the file for input/output, (use a 12 in the IOCB command) point to the LAST record, and read the last record. After the last record has been read, the system pointers will be looking at the end of the file, which is the obvious place to add your new record. Use NOTE, write the record, and add the key data and location to your key file. Always be sure to write your key file out to disk at the end of ANY session of using the Random Access routines.

How do we search the key file ?? If you remember, the key file is read into memory (or built in memory) and kept there while the data file is being manipulated. What appears to be an ideal length for me is 30 bytes per record. I have found that this can give me up to 3 keys to search on. Since each record is organized in the same manner as the rest, we have to build one routine that will allow us to test a given field of the record and compare against our search data which we input. Then, after finding the LENGTH of the entire key file, we step through it using a FOR NEXT loop that starts at ZERO and steps by 30 to access each key record individually. When we can then gosub to the routine which will convert the key record string data to useable numbers, POINT to the DATA RECORD, and retrieve it. The reason we don't kick out of the search loop yet is that there might be more than one record that might match our search data. (For instance, there were at one time 3 SUZUKIs, 2 DAVISs, and 2 LEEs all living here.) By not kicking out of the loop, we can easily look at all of the matching records and determine which we actually want. If you choose to kick out of the search loop upon finding any match, simply set the loop counter to the maximum value and NEXT LOOP. This will kick you out of the loop without losing the data from the last key record read, and you can then GOSUB or GO TO the routine which will retrieve the desired DATA RECORD.

All of this may look and sound complicated, but if you will try to work this out using the principles of structured programming, you will find that it is not that complex. Any program, no matter how complicated, is simply an organized collection of small routines, including random access data managers.

We're getting short on space, so we'll close for now. Next Month: working listings of the routines needed to build your own Random access program.

(Me, I'm gonna go eat somethin...)

BOOK REVIEWS

by Ron Ness

Our Eugene Public Library is not known for having a vast number of books on the subject of computers unless they concern the use of computers in the library. I have complained many times about this, and apparently it has done some good. My eagle-eyed wife found these goodies on the new book shelf.

'A BIT OF BASIC' by Thomas Dwyer and Margot Critchfield. c 1980 Addison-Wesley Publishing.

This book is primarily written for the Apple but unlike so many others, it does tell you what to do if your machine does not have a certain function they are using. There are better examples on using Atari's built in math functions than are in Atari's own Basic Reference Manual. It is worth your time to look for this book in your library.

'67 READY TO RUN PROGRAMS IN BASIC' by Wm. Scott Watson. c 1981 Tab Books \$12.95

This book is written for the TRS-80 but with some translation will run on the Atari. The programs are all 4K or less and cover graphics, home and business, education, and games. There are no sound routines and the graphics are limited to standard keyboard symbols. For the semi-advanced programmer this book will provide a starting place to expand your skills.

'THE A TO Z BOOK OF COMPUTER GAMES' by Thomas C. McIntire. c 1979 Tab Books \$7.95

The cover on this book says '26 exciting and instructive game programs ready to run on any computer' -- Well.....almost. There are 26 games and they are very instructive. As far as any computer, no. Some translation is necessary for the Atari, but anyone who has done some programming can make them work. For example, when the program calls for PRINT TAB(18);,,, you can use a position statement to get the same effect. The format of the entire book is building programs in modules. Lines 0 to 99 are always used to set up the game. Lines 100 to 999 are the main line of the program, lines 1000 to 1999 contain the instructions, and lines 2000,3000,etc. are the various subroutines called when the game is running. Each game has a flow chart and is very carefully documented. There are no sound routines and again the graphics are limited to standard keyboard symbols, but for those of you who are into sound and graphics, here is a goldmine of program ideas and a chance to unleash the power of your Atari.

'AN INVITATION TO PROGRAMMING 2, by Atari'. \$24.95

Writing Programs 1 consists of 7 lessons and a final quiz covering the keyboard, direct operating mode, mathematical precedence, variables (both string and numeric), if/then, goto, random numbers, and for/next. Writing Programs 2 has 8 lessons and a final quiz covering read/data, arrays, peek and poke, ASCII codes, string handling, and subroutines.

'AN INVITATION TO PROGRAMMING 3, by Atari'. \$24.95

Introduction to Sound is 8 lessons and a final quiz that cover sound effects, sound registers, & identifying notes, and multiple sounds. Introduction to Graphics has 7 lessons and a final quiz which cover the use of graphics modes 1 and 2, the use of the color registers, plot/drawto, changing modes, positioning of text material, and the special graphics characters. With each of these programs, Atari has thoughtfully provided a chart to write down the starting numbers on your cassette recorder for the start of the program load and start of the sound. By doing this, you can back up the tape to the start of the voice and review a lesson without having to reload the whole thing. This also helps when you get an error or bad load as I did on 3 occasions.

Since all of these lessons are geared to the 8K machines, the graphics demos leave something to be desired. The lessons themselves are quite good and you can cover them very rapidly. I found I was able to go over all 4 programs in about 4 hours. The final quiz will tell you which lesson you need to review. Twice thru each lesson and you should be able to score 100% on the final quiz.

While I feel \$50.00 is a bit spendy for these 4 programs for an individual, I do feel they are a DEFINITE MUST for your club library.

'CONVERSATIONAL SPANISH' by Atari. \$59.95

Atari has many fine programs. This is not one of them. Enough said.

Third World Atari

Revolutionary greetings from that land of spice and sun, Grenada. You've probably would have never heard of this little speck in the sea 20 miles by 10 miles if it were not for the political affiliation of the country with Cuba.

As a member of ACE, I thought I'd give you some idea of what it is like working with computers in a 3rd World country. If you are planning to bring a computer (hopfully an Atari) into a foreign country you DON'T want to say that it is a computer. If you do, chances are it will be taken for an indefinite period of time to determine if it has any military importance. For sure, leave the Missile Command at home !!!!!.

When we arrived the customs people did not know what we had - they thought the disc drive was an 8-track and the Atari 800 a typewriter. So be it. The duty for such items is 50% the cost, but then again typewriters and 8-tracks are a lot cheaper than computers. Since we brought our computer into the country last year, someone else brought one down and declared it as a personal computer. As I understand it, they have not gotten theirs back as of this writing.....

If you do need to declare what you are bringing into the country you have an advantage over other makes of computers. Atari is the type of computer that has a reputation of getting mixed up with video-toys, toasters and old fashioned washing machines. Internationally, Atari is known somewhat, and is connected with GAMES AND TOYS. So smile real big to the customs agents and let them know you have that Atari game thing that gets hooked up to the TV.

Electricity. You probably don't think much about it. It's there and it never goes out. Wrong. Many foreign countries (including Grenada) do load shedding. What this means is that you have electricity in the morning for a few hours and then it is turned off in your area and turned on in another. The schedule for this occurrence is non-existent so you back up every few screens or so. The "current" as it is known here, is not 110. It is usually not 220. Sometimes 280, sometimes 190. I've got a program to test the disc speed and find that I usually have to adjust it every day or so.

I have a surge sentry that everything runs through so I haven't lost or burnt out anything yet. This is IMPORTANT. People in the States will tell you that there is no difference between converters and transformers. Don't you believe it for one second. If you plug a TV into a converter and not a transformer you might as well kiss it good-bye. Can you guess what will happen if you plug your loving Atari 800 into a converter? Yes, you will see stars for a few seconds but your love affair will shortly come to a frazzled end.

Can you guess where you take a burnt out Atari is the West Indies? No where. You might as well take off the top cover, fill it with dirt and grow a mango tree in it!!!

No human person has the resources to fix in any length of time what it has taken you a parsec to do. What you can do (thanks to the kindly instructions of ACE masters) is to take the entire disc drive apart; clean, fix and adjust it on your own. A computer survival manual of self-repairs would be a welcome item for living in remote corners of the world.....

I'd welcome any personal communication on what's happening in the world of Atari.

Lint Hutchinson
Library Office
St. George's University School of Medicine
St. George, Grenada
WEST INDIES

Send a business-size SASE to the Ross' for the new, updated ACE Library List!!

Best of ACE-1981, still available, Disk or Tape, \$8!

Your Atari Computer
a guide to the Atari 400/800 personal computers
by Lon Poole
Osborne/McGraw Hill (\$15)
review by M.R. Dunn, Co-Editor

This is the book we have all been waiting for- the most comprehensive guide yet for the Atari computer owner. Excellent for both the beginner and the advanced Atari owner, this 458 page book is crammed full of useful information. The first few chapters are clearly written and illustrated step-by-step instructions on how to hook up your computer and all the available peripherals, including an excellent tutorial on the use of the disk drive. About half the pages are instructions on how to use the hardware; the rest of the book is on programming in Atari BASIC. Not only explaining each keyword clearly, there are many useful subroutines illustrated and explained, including various ones decimal align or right justify output, print mailing labels, make a cassette based mailing list, formatting and error-proofing inputs; all clearly explained so you can use them in your own programs.

The chapters on I/O, beginning and advanced graphics including character set animation are great. There are even 9 useful appendixes that are almost worth the price of the book! I would recommend this book highly to all owners especially new ones; in fact, I think the dealers should consider giving one with each system. The saving answering the questions of their customers would quickly pay for the book.

Fileit 2+ update
(Swift Software, POB 641, Melville, NY 11747 \$50)

In the last issue, Fileit was reviewed by Stacy Goff, and some suggestions were made on improving the program. While our newsletter was at press, I received the following from Jerry White, the author of Fileit 2. In talking to Jerry on the phone, many of the suggestions of Stacy were already taking care of, especially a vast speed up in the sort. The new version is called Fileit 2+, it is available from Swift now. Jerry would like to emphasize that this is the only one that is non-protected so the user can modify the program as he wishes.

M.R. Dunn

Summary of changes
by Jerry White

The Fileit 2 package has been updated with many changes as a result of user suggestions. There has been a great speed up in operations; for example, there is very little hesitation between records when they are added. There are two new programs included, SORT and FINSORT, that are machine language sort programs. These are loaded in when the disk is booted, and greatly increase the speed of sorting. The FINSORT program provides machine speed sort of up to 6 fields on financial data files. Error handling has been improved also. The new version is called FILEIT 2+. (Jerry mentions some other new features that would be mainly to use to present owners).

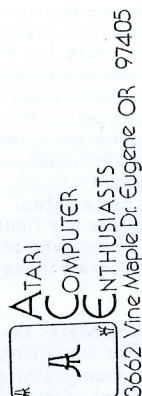
QUESTIONNAIRE
from Jerry White

Please answer a few questions for me to help to establish the wants and needs of Atari owners, as well as provide subject matter for future articles.

- 1) I write tutorials and reviews for newsletters-what would you like me to write about?
- 2) If you could have a program written to your specifications, what would the program do?
- 3) List the three software products you use most often.
- 4) Do you have any questions about Atari BASIC that you would like to see answered in a future article?
- 5) If you have any of my software, let me know what you think about it.

Please indicated the name of your users group at the top of the page. If you would like a copy of my personal Atari Software catalog, or answers to questions, please include a SASE.

I'll be looking forward to hearing from you. Jerry White, 18 Hickory Lane, Levittown, NY 11756.



Atari Computer Enthusiasts

A.C.E. is an independent computer club and user's group with no connection to the Atari Company, a division of Warner Communication Company. We are a group interested in education our members in the use of the Atari Computer and in giving the latest News, Reviews and Rumors.

All our articles, reviews and programs come from you, our members.

Our membership is world-wide in scope; membership fees include the A.C.E. Newsletter. Dues are \$10 a year for U.S., and \$20 a year Overseas Airmail.

President-Kirt Stockwell, 1810 Harris #139, Eugene, Or 97403 503-686-2470

Secretary-Charles Andrews, POB 1613, Eugene, Or 97401 503-747-9892

Librarian-Chuck and Jody Ross, 2222 Ironwood, Eugene, Or 97401 503-343-5545

Editors -Mike Dunn, 3662 Vine Maple Dr., Eugene, Or 97405 503-344-6193
-Jim Burgess, 1405 N. 26th Ave, Eugene, Or 97401 503-484-9925

**FIRST
CLASS
MAIL**

Handwritten signature